

INTEGRATED CIRCUIT CHIP

Patent number: JP2001154873
Publication date: 2001-06-08
Inventor: DAVID ALAN EDWARDS; RICH ANTHONY WILLIS
Applicant: HITACHI LTD
Classification:
 - international: G06F11/28; G06F15/78
 - european:
Application number: JP20000292798 20000926
Priority number(s):

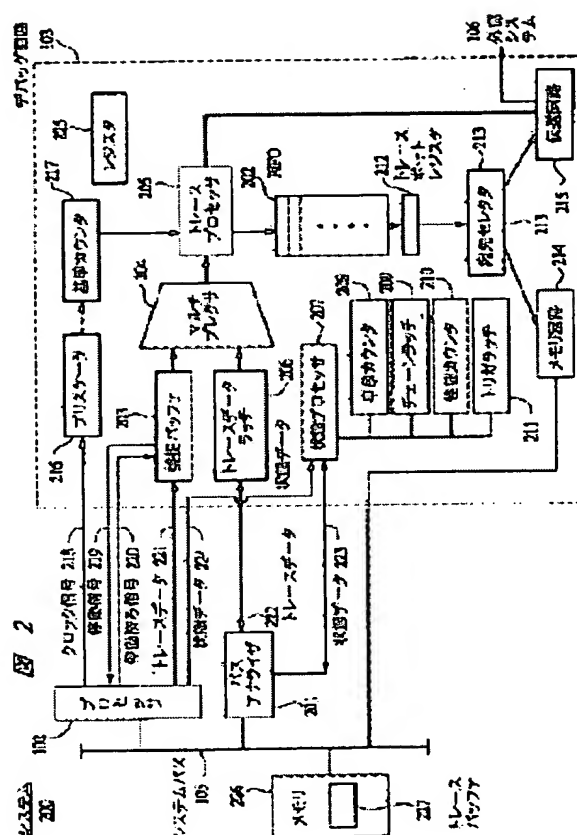
Also published as:

US6684348 (B)

Abstract of JP2001154873

PROBLEM TO BE SOLVED: To provide a trace system without any interruption which receives trace information from one or more processors or from the other device.

SOLUTION: A trace system 200 can be constituted by a user so as to be operated in various modes so that trace information is flexibly stored or transmitted. This trace system is provided with a memory-mapped FIFO 202 to which access can be performed without affecting processor performance. In one configuration, this trace system is provided with a trace buffer 227 for receiving trace information at the inside clock speed of the processor 102. In another configuration, a compression protocol is imparted so that the trace message on-chip can be compressed prior to the message transmission to an outside system 106 or to the storage of the message in the memory.



Data supplied from the esp@cenet database - Worldwide

Best Available Copy

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2001-154873

(P2001-154873A)

(43) 公開日 平成13年6月8日 (2001.6.8)

| (51) Int.Cl. ⁷ | 識別記号 | F I | テマコード(参考) |
|---------------------------|-------|---------------|-----------|
| G 0 6 F 11/28 | 3 1 0 | G 0 6 F 11/28 | 3 1 0 B |
| 15/78 | 5 1 0 | 15/78 | 5 1 0 K |

審査請求 未請求 請求項の数40 O L (全 28 頁)

(21) 出願番号 特願2000-292798(P2000-292798)
(22) 出願日 平成12年9月26日(2000.9.26)
(31) 優先権主張番号 09/409612
(32) 優先日 平成11年10月1日(1999.10.1)
(33) 優先権主張国 米国 (U S)

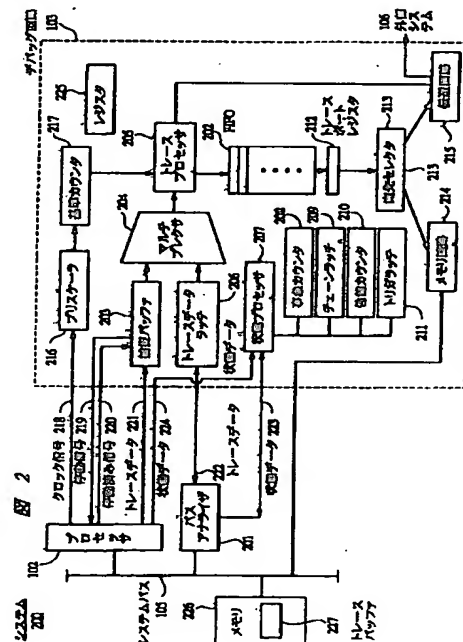
(71) 出願人 000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目6番地
(72) 発明者 デビッド・アラン・エドワーズ
イギリス国、プリストル BS8 1L
B、クリフトン、プロ・キャセドラル・レ
ーン 6
(72) 発明者 アンソニー・ウィリス・リッチ
ニュージーランド国、ケンブリッジ、
2351、P. O. Box 761、クララ・ス
トリート 40
(74) 代理人 100080001
弁理士 筒井 大和

(54) 【発明の名称】 集積回路チップ

(57) 【要約】

【課題】 1つ又は複数のプロセッサ、又は他のデバイスからトレース情報を受け取った割込なしトレースシステムを提供する。

【解決手段】 トレースシステム200は、トレース情報を柔軟に記憶又は伝送するために種々のモードで作動するように、ユーザによって構成可能である。トレースシステムは、メモリマップされ、プロセッサ性能に影響を及ぼすことなくアクセス可能なFIFO202を含む。一態様では、トレースシステムは、プロセッサ102の内部クロック速度でトレース情報を受け取るトレースバッファ227を含む。別の実施の態様では、外部システム106へメッセージを伝送し、又は、メモリ内にメッセージを記憶するのに先立って、圧縮プロトコルが、トレースメッセージオンチップを圧縮するために与えられる。



【特許請求の範囲】

【請求項 1】 集積回路チップであって、
少なくとも 1 つのプロセッサと、
該プロセッサに動作的に接続され、先入れ先出し（F I F O）バッファを含み、前記プロセッサで発生されたトレース情報を記憶するようにユーザによって構成可能であるデバッグモジュールと、
トレース情報オフチップを伝送するインタフェースとを備え、

前記トレース情報が圧縮フォーマット内に記憶され、オフチップ伝送されることを特徴とする集積回路チップ。

【請求項 2】 請求項 1 記載の集積回路チップであって、

前記トレース情報が、前記 F I F O バッファ内の前記情報を記憶する前に、バッファリングされることを特徴とする集積回路チップ。

【請求項 3】 請求項 1 記載の集積回路チップであって、

前記トレース情報が、前記 F I F O バッファ内の前記情報を記憶する前に、フィルタリングされることを特徴とする集積回路チップ。

【請求項 4】 請求項 1 記載の集積回路チップであって、

前記トレース情報が以下のうち少なくとも 1 つを選択的に伝送することを特徴とする集積回路チップ；前記プロセッサに組合された記憶回路と、
前記デバッグモジュールに対して外部のデバッグシステムに接続されたリンク。

【請求項 5】 請求項 4 記載の集積回路チップであって、

前記 F I F O が満杯状態に接近した時に、前記デバッグモジュールが、前記プロセッサを停動するため、制御信号を前記プロセッサに伝送することを特徴とする集積回路チップ。

【請求項 6】 請求項 5 記載の集積回路チップであって、

前記 F I F O が満杯状態に接近するという指示を、前記デバッグモジュールが前記外部デバッグシステムに与えることを特徴とする集積回路チップ。

【請求項 7】 請求項 5 記載の集積回路チップであって、

前記プロセッサの実行パイプラインが停動したという指示を、前記デバッグモジュールが前記外部デバッグシステムに与えることを特徴とする集積回路チップ。

【請求項 8】 請求項 5 記載の集積回路チップであって、

前記デバッグモジュールが、前記 F I F O の満杯時にトレース情報を廃棄するようなトレース保持モードで構成されていることを特徴とする集積回路チップ。

【請求項 9】 請求項 1 記載の集積回路チップであって、

て、

前記トレース情報では、前記デバッグモジュールが、1 つ又はそれ以上のトレースメッセージを廃棄したという指示を伝送することを特徴とする集積回路チップ。

【請求項 10】 請求項 7 記載の集積回路チップであって、

前記プロセッサが停動された前記指示が、トレースメッセージ内に含まれることを特徴とする集積回路チップ。

【請求項 11】 請求項 1 記載の集積回路チップであって、

前記集積回路が、以下を含んだ一集団のステップのうち少なくとも 1 つによって前記トレース情報を圧縮する回路を含むことを特徴とする集積回路チップ；オペランドアドレス、プログラムカウンタ、バスアナライザ、及び、命令アドレスのうち少なくとも 1 つによって表示し、時間差によってタイミング情報を表示する。

【請求項 12】 請求項 11 記載の集積回路チップであって、

前記プロセッサが以下のうち少なくとも 1 つを含んだ標準メッセージを送ることを特徴とする集積回路チップ；プロセッサ処理の時間状態を表示するタイミング情報と、

前記プロセッサのプログラムカウンタと、
システムバスの装置のアドレス。

【請求項 13】 請求項 11 記載の集積回路チップであって、

前記プログラムカウンタが、プロセッサのシャドープログラムカウンタであることを特徴とする集積回路チップ。

【請求項 14】 請求項 11 記載の集積回路チップであって、

前記トレース情報が、コンピュータシステムのシステムバスに作動的に接続された回路から生成されることを特徴とする集積回路チップ。

【請求項 15】 請求項 12 記載の集積回路チップであって、

前記プロセッサが、時間間隔の制限数内で標準メッセージを生成することを特徴とする集積回路チップ。

【請求項 16】 請求項 12 記載の集積回路チップであって、

デバッグインタフェースが時間間隔の特定数以上に休止した場合に、前記標準メッセージが生成されることを特徴とする集積回路チップ。

【請求項 17】 請求項 12 記載の集積回路チップであって、

前記タイミング情報がプロセッサクロックから得られることを特徴とする集積回路チップ。

【請求項 18】 請求項 11 記載の集積回路チップであって、

前記時間差が、最新組から以前組までのトレース情報の

時間差であることを特徴とする集積回路チップ。

【請求項19】 請求項11記載の集積回路チップであって、
アドレスオフセットが、新しいアドレスから以前に伝送されたアドレスを引くことによって決定されることを特徴とする集積回路チップ。

【請求項20】 請求項11記載の集積回路チップであって、
前記トレース情報が、アドレス情報とタイミング情報との少なくとも1つの絶対又は相対符号化を使って符号化されたことを特徴とする集積回路チップ。

【請求項21】 集積回路チップであって、
少なくとも一つのプロセッサと、
該プロセッサに動作的に接続され、先入れ先出し（FIFO）バッファを含み、前記プロセッサで生成されたトレース情報を記憶するように、ユーザによって構成可能であるデバッグモジュールと、
トレース情報オフチップを伝送する伝送手段とを備え、
前記トレース情報が、圧縮フォーマット内に記憶され、オフチップ伝送されることを特徴とする集積回路チップ。

【請求項22】 請求項21記載の集積回路チップであって、
前記デバッグモジュールが、前記FIFOバッファに前記情報を記憶する前に、トレース情報をバッファリングする手段を含むことを特徴とする集積回路チップ。

【請求項23】 請求項21記載の集積回路チップであって、
前記デバッグモジュールが、前記FIFOバッファ内の前記情報を記憶する前に、前記トレース情報をフィルタリングする手段を含むことを特徴とする集積回路チップ。

【請求項24】 請求項21記載の集積回路チップであって、
前記集積回路が、以下の少なくとも1つに前記トレース情報を選択的に伝送する手段を含むことを特徴とする集積回路チップ；前記プロセッサと関連された記憶回路と、
前記デバッグモジュールに対して外部のデバッグシステムに接続されたリンク。

【請求項25】 請求項21記載の集積回路チップであって、
前記集積回路が、前記トレース情報を圧縮する圧縮手段を有し、
前記圧縮手段が以下の少なくとも1つを含むことを特徴とする集積回路チップ；オペランドアドレス、プログラムカウンタ、バスアナライザ、及び、命令アドレスのうち少なくとも1つを、少なくとも1つの前記アドレスに関して符号アドレスオフセットとして表示する手段と、
時間差でタイミング情報を表示する手段。

【請求項26】 請求項25記載の集積回路チップであって、

前記プロセッサが、以下のうち少なくとも1つを含む基準メッセージを送ることを特徴とする集積回路チップ；
プロセッサ処理の時間状態を表示するタイミング情報と、
前記プロセッサのプログラムカウンタと、
システムバスにおける装置のアドレス。

【請求項27】 請求項25記載の集積回路チップであって、

前記プログラムカウンタがプロセッサのシャドープログラムカウンタであることを特徴とする集積回路チップ。

【請求項28】 請求項25記載の集積回路チップであって、

前記トレース情報が、コンピュータシステムのシステムバスに作動的に接続されたアナライザから生成されることを特徴とする集積回路チップ。

【請求項29】 請求項26記載の集積回路チップであって、

前記プロセッサが、時間間隔の所定数内に基準メッセージを生成することを特徴とする集積回路チップ。

【請求項30】 請求項26記載の集積回路チップであって、

デバッグインタフェースが時間間隔の特定数以上に休止した場合に、前記基準メッセージが生成されることを特徴とする集積回路チップ。

【請求項31】 請求項26記載の集積回路チップであって、

前記タイミング情報がプロセッサクロックから得られることを特徴とする集積回路チップ。

【請求項32】 請求項25記載の集積回路チップであって、

前記時間差が、最新組から以前組までのトレース情報の時間差であることを特徴とする集積回路チップ。

【請求項33】 請求項25記載の集積回路チップであって、

アドレスオフセットが、新しいアドレスから以前に送られたアドレスを引くことによって決定されることを特徴とする集積回路チップ。

【請求項34】 請求項25記載の集積回路チップであって、

前記トレース情報が、アドレス情報とタイミング情報とのうち少なくとも1つの絶対又は相対符号化を用いて符号化されることを特徴とする集積回路チップ。

【請求項35】 請求項24記載の集積回路チップであって、

前記FIFOが満杯状態に接近した時に、前記デバッグモジュールが、前記プロセッサを停動するために制御信号を前記プロセッサに伝送することを特徴とする集積回路チップ。

【請求項36】 請求項35記載の集積回路チップであって、

前記FIFOが前記満杯状態に接近するという指示を、前記デバッグモジュールが前記外部デバッグシステムに与えることを特徴とする集積回路チップ。

【請求項37】 請求項35記載の集積回路チップであって、

前記デバッグモジュールが、前記プロセッサにおける実行パイプラインの停動の指示を前記外部デバッグシステムに与えることを特徴とする集積回路チップ。

【請求項38】 請求項35記載の集積回路チップであって、

前記デバッグモジュールが、前記FIFOの満杯時にトレース情報を廃棄するようなトレース保持モードで構成可能であることを特徴とする集積回路チップ。

【請求項39】 前記21記載の集積回路チップであって、

前記トレース情報では、前記デバッグモジュールが、1つ又はそれ以上のトレースメッセージを廃棄したという指示を伝送することを特徴とする集積回路チップ。

【請求項40】 請求項37記載の集積回路チップであって、

前記プロセッサが停動されたという前記指示が、トレースメッセージ内に含まれることを特徴とする集積回路チップ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は一般に、システムオンチップ上でトレースを実行することに関し、更に詳細にはトレース情報を記憶する装置及び方法に関する。

【0002】

【従来の技術】システムオンチップデバイス(SOC)はよく知られている。このデバイスには一般に1つのプロセッサ、1つ又は複数のモジュール、複数のバスインタフェース、複数のメモリデバイス、及び、情報通信用の1つ又は複数のシステムバスが含まれる。複数のマルチモジュール、及びそれらの通信がチップにとって内部的に発生するので、ソフトウェア又はハードウェア内で問題が発生したとき、一般に、この情報へのアクセスが困難になる。従って、これらのシステム上でデバッグする(誤りを見つけて取り除く)ことは簡単ではない。これらSOCの開発結果として、チップにおける性能及びトレース情報を監視するために、特殊なデバッグシステムが開発された。そのようなシステムは一般に、シリアル通信を介してプロセッサにアクセスするデバッグツール及びデバッグソフトウェアのような専用のハードウェア及びソフトウェアを含む。

【0003】しかしながら、SOCをデバッグすることは一般に、1つ又は複数のプロセッサレジスタ、又は記憶場所を割込的に監視することを伴う。記憶場所へのア

クセスは時には破壊的であり、デバッグツールから読み取られるべき場所へのデータアクセスはプロセッサ性能を妨害することもあり得る。同様に、複数のアクセスは一般にシステムバスを介してプロセッサ、メモリ、又は他のモジュールに対して実行され、一般的なオペレーションを実行するために当該システムバスを介して利用可能な帯域幅を減少させることも可能である。幾つかのデバッグシステムは、プロセッサと同じクロック速度で機能せず、例えばトレース情報の獲得のようなデバッグ機能の使用を可能にするために、当該プロセッサの動作を減速することが必要なこともあり得る。プロセッサを休止又は減速させることによって、幾つかのエラーのタイプが再生不能になり、従って、検出又は補正不可能になる。更に、プロセッサの高速に起因して、精確な情報が全く利用不可能というわけではない。即ち、情報が歪められるか、又は失われることもあり得る。

【0004】あるシステムはSOC内に1つ又は複数の専用機能ユニットを含み、その専用機能ユニットは当該プロセッサのデバッグの専用であり、時にはデバッグユニット又はモジュールと呼ばれる。しかしながら、例えばトレース情報のような情報入手の際に、これらのユニットは当該プロセッサのオペレーション(動作)に影響を及ぼす。これらのデバイスは、一般に当該プロセッサよりも低速で機能し、従って、プロセッサデータへのアクセスの際に、プロセッサのオペレーションに影響を及ぼす。デバッグシステムは、ターゲットプロセッサ自体の上でデバッグコードを実行するのに頼り、このコードが通常デバッグされる方に組み込まれる。従って、デバッグコードの存在は、メモリ設計の観点から割込的であり、命令ストリームの混乱になる。

【0005】回路内エミュレータ(ICE)と呼ばれる他のデバッグシステムは、オンチップハードウェアにマッチし、それに接続される。従って、オンチップ接続部はエミュレータ上でマップされ(見つけ出され)、エミュレータ上でアクセス可能である。しかしながら、エミュレータは、あるアプリケーションに対して法外に高価であり、全てのオンチップ速度、又は通信に首尾よくマッチしていない。従って、エミュレータシステムは不適当である。更に、これらのシステムは通常、システムバスを介して情報を転送し、それによって必然的にプロセッサ性能に影響を与える。

【0006】故障修理用の別の技術は論理状態アナライザ(LSA)を使用することを含み、LSAは集積回路のピンに接続されたデバイスであり、その集積回路は全てのオフチップ通信の状態を監視する。LSAデバイスは一般に高価なデバイスであり、LSAデバイスによってアクセスが当該チップ内の情報を釘付け不可能にする。

【0007】

【発明が解決しようとする課題】要するに、プロセッサの内部状態を監視するために、そして割込ない方法でリアルタイム状態及びリアルタイムトレースのような機能を提供するために不適當な多くのシステムが存在する。

【0008】従来のデバッグシステムにおけるこれら及び他の欠点は、1つ又は複数のプロセッサ、又は他のデバイスからトレース情報を受け取った割込なしトレースシステムを提供することによって回避される。

【0009】

【課題を解決するための手段】トレースシステムは、トレース情報を記憶する先入れ先出し(FIFO)バッファを含み、そのFIFOバッファはメモリマップされ、プロセッサ性能に影響されることなく他のシステムによってアクセス可能である。本発明の一態様では、トレースシステムはプロセッサの内部クロック速度で情報を受け取るトレースバッファを含む。

【0010】トレース情報を直接的にどのように処理し、及び、プロセッサオペレーションでどのように割り込むかを明確に述べた割込なし方法が提供される。詳細には、トレースシステムはデバイスを含み、そのデバイスはプロセッサから分離して作動し、プロセッサの内部クロック速度で作動し、又は、トレース情報のロスがプロセッサオペレーションに影響しないようなモードで作動する。更に、収集されたトレース情報は、トレース動作を実行するのに要する全ての情報を含み、プロセッサは、例えばデバッグツール上で実行したソフトウェアプログラムによって追加情報を入手するために、割り込む必要がない。本発明の一態様によれば、トレースシステムは、プロセッサと関連し、ソフトウェアの介入を必要としないハードウェア内で実行される。一態様では、トレース情報はアドレス情報とメッセージ情報との両方を含む。別の態様では、トレース情報はタイミング情報を含む。

【0011】別の態様において、トレース情報はトレースシステムによって圧縮される。情報を圧縮することによって、トレース情報は、低帯域幅リンクを介して伝送するために保存され、オンチップトレース記憶を最大化する。例えば、トレース情報は、タイムスタンプ及びアドレス情報を圧縮することによって圧縮可能である。更に、トレース情報は、例えば特定のオペレーションタイプにおける1つのトレースパケットのように、重複タイプの情報を省略することによって圧縮可能である。また、情報は、トレース情報を生成する規準を予め決定することによってフィルタ可能である。情報をフィルタし、重複の情報オンチップを除去することにより、外部システムへのリンクの帯域幅要件と、オンチップ記憶要件とが軽減される。トレースメッセージは、メッセージ内のデータフィールドの絶対値を相対値に取り替えることによって、圧縮可能である。圧縮情報の絶対値を含んだ周期的な基準メッセージは、相対値に基準を提供する

ように設けられる。

【0012】これらの及びその他の利点は集積回路によって提供され、その集積回路は、少なくとも1つのプロセッサと、プロセッサに作動的に接続され、先入れ先出し(FIFO)バッファを含んだデバッグモジュールとを備え、デバッグモジュールがユーザによって構成されて、以下のモードのうち少なくとも1つでプロセッサによって生成されたトレース情報を記憶する；FIFOバッファが循環バッファとして構成されるモードと、FIFOバッファの満杯時に、デバッグモジュールがトレース情報の記憶を停止するモードと、FIFOバッファの満杯時にデバッグモジュールが追加トレース情報を廃棄するモード。

【0013】一態様において、デバッグモジュールは、FIFOが満杯レベルへ接近しているという指示を、プロセッサに与える。例えば、デバッグモジュールは、FIFOの満杯レベルから記憶場所の制限数内に指示を与える。一態様では、デバッグモジュールが、FIFOバッファの満杯時に、トレース情報の記憶を停止するようなモードにおいて、集積回路が、トレース情報の記憶を又はトレース情報の生成を作動不能にするような割込を生成する。また、集積回路が、割込を受け入れ、トレース情報の記憶を又はトレース情報の生成を作動不能にし、追加トレース情報を記憶可能にするFIFOを空にするハンドラコードを実行する。その代りに、ユーザがFIFOのコンテンツを手動で検査可能にするソフトウェアシステムを提供できる。一態様では、トレース情報はアドレス及びメッセージ情報を含む。

【0014】一態様では、トレース情報がバイトベース境界上に記憶される。一態様では、FIFOに記憶されるトレース情報が、可変長なメッセージを含む。また、トレース情報が、プロセッサのトリガされた状態を表す複数の状態情報ビットを含む。

【0015】一態様では、FIFOの記憶場所が、メモリマップ記憶レジスタである。

【0016】本発明の一態様では、FIFOが、プロセッサに作動的に結合された捕捉バッファからトレース情報を受け取る。一態様では、捕捉バッファが、システムバスから分離したバスによってプロセッサと他のトレース生成装置とに作動的に結合される。

【0017】一態様では、デバッグモジュールが、新しい命令の実行を停動するために、制御信号をプロセッサに与える。一態様では、デバッグモジュールが制御信号を供給し、捕捉バッファの満杯レベルから記憶場所の制限数内で、記憶レベルに到達した捕捉バッファに応答してプロセッサを停動するようにした。一態様では、デバッグモジュールが制御信号を供給し、FIFOの満杯レベルから記憶場所の制限数内で、記憶レベルに到達したFIFOに応答してプロセッサを停動するようにした。一態様では、記憶場所の制限数が、停動されたプロセッ

サによって生成可能なトレースメッセージ数以上である。

【0018】本発明の別の態様によれば、集積回路は、少なくとも1つのプロセッサと、プロセッサに動作的に接続され、先入れ先出し（FIFO）バッファを有するデバッグモジュールとを備えて提供され、デバッグモジュールが、プロセッサによって生成されたトレース情報を記憶するように、ユーザーによって構成され、FIFOが、トレース情報を保持するように構成された複数のメモリマッピングレジスタを有する。

【0019】一態様では、FIFOが、満杯時にトレース情報を廃棄するようなモードで動作する。一態様では、デバッグモジュールが、FIFOが満杯レベルへ接近しているという指示を、プロセッサに与える。また、デバッグモジュールが、FIFOの満杯レベルから記憶場所の制限数内に、指示を与える。

【0020】一態様では、メモリマッピングレジスタが、プロセッサから独立した読み出し回路によって読み出される。他の態様によれば、システムバスを更に有する集積回路には、トレース情報が、システムバスから離れた通信リンクを介して伝送される。

【0021】一態様では、トレース情報が、プログラムカウンタ情報と、プロセス認識情報と、アドレス情報と、メモリ内に記憶された情報と、システムバストランザクションに関連した情報とのうち少なくとも1つを有する。

【0022】一態様では、トレース情報が、RAMベース記憶ユニット内に記憶される。

【0023】一態様では、トレース情報が、RAMベースFIFO及びFIFOバッファ内に記憶される。

【0024】本発明の他の態様によれば、集積回路チップは、少なくとも1つのプロセッサと、プロセッサに動作的に接続され、先入れ先出し（FIFO）バッファを含み、プロセッサで発生されたトレース情報を記憶するようにユーザーによって構成可能であるデバッグモジュールと、トレース情報オフチップを伝送するインタフェースとを備えて提供され、トレース情報が圧縮フォーマット内に記憶され、オフチップ伝送される。一態様では、トレース情報が、プロセッサに組合された記憶回路と、デバッグモジュールに対して外部のデバッグシステムに接続されたリンクとのうち少なくとも1つを選択的に伝送する。一態様では、トレース情報は、1つ又はそれ以上のトレースパケットがロスしたという指示を含む。別の態様では、トレース情報を生成している間にプロセッサが停動されたという指示を、トレース情報が含む。

【0025】一態様では、集積回路は電気回路の構成部分を含み、その電気回路の構成部分は、オペランドアドレス、プログラムカウンタ、バスアナライザ、及び、アドレス命令のうち少なくとも1つを、その少なくとも1つのアドレスに関係する符号付きオフセットとして表示

し、そして時間差によってタイミング情報を表示することによりトレース情報を圧縮する。一態様では、アドレスオフセットは、新しりアドレスから以前に送られたアドレスを引くことによって決定される。

【0026】本発明の一態様では、コンピュータシステムのトレース情報を圧縮する方法が提供され、その方法が、相対アドレスとしてオペランドアドレス、プログラムカウンタ、バスアナライザ、及び、命令アドレスのうち少なくとも1つを表示し、時間差でコンピュータシステムのタイミング情報を表示するステップを有する。

【0027】一態様では、トレース情報が少なくとも1つのプロセッサで生成され、方法が、プロセッサ処理の時間状態を表示するタイミング情報と、プロセッサのプログラムカウンタと、システムバスにおける装置のアドレスとのうち少なくとも1つを含んだ標準メッセージを送るステップを有する。本発明の別の態様では、回復方法（decompression method）が、時間差からの絶対情報及び／又は相対アドレス情報と、以前のトレース情報内で受け取られた情報とを受け取って提供される。

【0028】一態様では、トレース情報用の見出し情報を生成するアドレス情報とタイミング情報とのうち少なくとも1つの絶対又は相対符号化であって、見出し情報がトレース情報におけるトレース情報ソースのタイプを含み、メッセージにおけるソースの情報が、メッセージを生成するソースモジュールの指示と、そのソースモジュールのチャンネルの指示とのうち少なくとも1つを含む。一態様では、見出し情報は、アドレス情報が絶対アドレス情報又は相対アドレス情報であるか否かの表示を含む。一態様では、相対アドレスは符号付きオフセットアドレスである。一態様では、タイミング差情報は、最終基準トレースメッセージが生成されてから、多数のタイマーインクリメントを表す。一態様では、トレース情報は、コンピュータシステムのメモリに書き込まれたデータを含む。

【0029】本発明の更なる特徴及び利点、並びに本発明の様々な態様における構造及び動作（オペレーション）は、添付図面を参照しながら、以下に詳細に説明される。図面において、同じ参照番号は同等又は機能的に類似の要素を示す。その上、参照番号における左側の一桁又は二桁の数字は、その参照番号が最初に現れた図面を識別する。

【0030】本発明は、添付された特許請求の範囲において詳細に指摘される。本発明の前述及び更なる利点は、類似の参照番号が同一又は類似の要素を表示する添付図面と共に以下の記述を参照することによって更によく理解されだろう。

【0031】

【発明の実施の形態】本発明の一実施の形態が図1に関して詳細に説明される。図1は集積回路デバイス101、即ち上述のシステムオンチップ（SOC）のプロッ

ク図を示す。この回路は、システムバス105で接続されたプロセッサ102とデバッグ回路103を含む。システムバスは、従来のバス、バケットスイッチ、又は、他の通信媒体であってもよい。他の通信媒体は、デバイス101のモジュールの間でオペレーティング情報を通信するために用いられる。例えば、読み、書き、スワップ等のオペレーションは、モジュールの間で実行される一般的オペレーションである。

【0032】プロセッサ102は、1つ又は複数のプロセッサ命令内のプログラムコードを読み込んで実行し、そしてデータに基づいてオペレーションを実行するように変更されたデバイスである。プロセッサ102は、多数のデータソースからデータを読み出し、1つ又は複数のデータ記憶装置（図示せず）にデータを書き込む。これらのデータ記憶装置は、ランダムアクセスメモリ（RAM）、ディスクコントローラを介してアクセス可能なコンピュータハードディスク、1つ又は複数の通信リンクを介してアクセス可能な記憶装置、又は、データの記憶用の幾つかのエンティティを含む。これらの記憶エンティティはシステムバス105上で直接アクセス可能であり、又は外部通信リンクを介してアクセス可能である。

【0033】通信リンク104は、プロセッサ102をデバッグ回路103に結合し、本発明の様々な実施の形態に従って、システムバス105から分離される。リンク104は、デバッグ情報をプロセッサ102からデバッグ回路103へ転送し、状態及びプロセッサ制御情報をデバッグ回路103からプロセッサ102へ転送するように構成される。デバッグ回路103が1つ又は複数のプロセッサ102、又は他のデバイスに結合され得ることが理解される。デバッグ回路103はマルチプレクサデバイスを介して1つ又は複数のプロセッサへ結合され、そのマルチプレクサデバイスは、例えば、状態及び制御情報をプロセッサへ多重通信し、且つ状態及び制御情報をプロセッサから多重通信する。

【0034】本発明の一形態では、プロセッサ102は、リンク104を介してデバッグ回路103にトレース情報を供給し、プロセッサ102又はシステムバス105の性能に影響しない方法でこの情報を供給する。その代わりに、トレース情報は、システムバス105に添付された1つ又は複数のモジュールによって生成され、モジュールはプロセッサ、インタフェースモジュール、バスコントローラ、バスアナライザ、又は、トレース情報の生成可能な任意のモジュールが含まれる。トレース情報はデバッグ回路103によって受け取られ、そこでは処理及び記憶され、又は外部システム106に転送される。

【0035】一般に、極く限られたトレース情報は、割込なし、即ち、例外又は割込を備えたプロセッサを混乱させることなく収集可能である。様々の実施の形態によ

れば、デバッグ回路は、収集可能なトレース情報量を最大化しようとする。

【0036】割込なく収集されるトレースデータ量を最大化するために、幾つかの技術を使用できる。

【0037】トレース情報オンチップを記憶し、又は、トレース情報オフチップを送る以前に情報を圧縮する。例えば、命令アドレス、オペランドアドレス、及びタイムスタンプ情報が圧縮されて記憶空間を節約し、且つ、外部システム106へのトレース情報の転送に必要な帯域幅の量を減少させる。

【0038】プロセッサを停動することなくトレースする。

【0039】もしどの地点でトレースデータが失われたかを決定可能であれば、幾らかのトレース情報は失われるかもしれないが、そのことはやむなく容認される。デバッグシステムは、失われたメッセージの指示を外部システム106（即ちデバッグツール）に、例えばトレースメッセージ自体内に、又は、信号を送ることを通して、供給可能である。トレースメッセージ内に情報を含むことによって、トレース情報が失われた地点が決定可能である。

【0040】プロセッサを停動した状態でトレースする。例えばトレースバッファが満杯の時、又は、他の資源が利用不可能である時のように、トレースシステムが追加トレース情報を受け入れ不可能である時に、このタイプのトレースは、プロセッサを停動することを伴う。このトレースの体系によって精確なトレースデータが所定の地点に寄せ集め可能となる。制限時間の間に全てのトレース情報をセーブすることが重要である時に、このタイプのトレースが決定的になる。更に、もしデバッグ回路内の資源が利用不可能であるならば、トレース情報が保存されるように、デバッグ回路が制御信号を供給してプロセッサを停動する。

【0041】その次に、デバッグ回路がデバッグ割込を生成し、ハンドラコードを実行してトレース情報の収集を不可能にし、デバッグ回路によってプロセッサが影響なく進行する。その代わりに、トレース情報を生成する回路を作動不能にする（割込を抑制する）ことも可能である。デバッグ回路はデータを抽出し、トレース情報の収集を開始することができる。別の実施の形態では、デバッグ回路がトレースバッファからトレースデータを抽出し、プロセッサが一時停止され、そしてプロセッサが再開可能となる。また、トリガ信号をデバッグ回路から外部システムに提供可能である。外部システムは、トレースバッファ又はFIFOの満杯又は略満杯であるようなトレース関連状態を示し、又は、満杯である又は満杯になるトレースバッファ又はFIFOのために、プロセッサを停動することを示す。外部システムがトレースバッファ又はFIFOを空にするべきであり、又は、他の修正措置を採用するべきであることを、外部トリガが示

す。

【0042】更に、生成されたトレース量を減少させるように、トレース情報をインテリジェント的に収集可能である。トレースの抽出の時、通常、特定の関心項目がある。例えば、次のようである。

【0043】コールグラフプロファイリング（呼出グラフィメージ）用にトレースした分岐（ブランチ）を実行する時に、単なるサブルーチン分岐とサブルーチン分岐からのリターンとが興味対象である。

【0044】アプリケーション内のコード又はデータフローをトレースした時に、コード又はデータフローが通常、一回で単一プロセッサ処理用に考えられている。

【0045】分岐トレース、即ちデータトレース分析は一回で単一のコードレンジ又は機能内で時には考察される。

【0046】例えばロジックアナライザ及び外部トレースバッファシステムのような従来の外部システム106は一般に、全てのトレース情報を外部へ抽出し、それから外部システム106自身内のトレース情報をフィルタする必要がある。従って、従来のトレース説明システムはオフチップ通信帯域幅を十分に利用していない。本発明の様々な実施の形態によれば、トレースがフィルタされ、選択的に生成され、圧縮されてオンチップのトレースデータを記憶するために要求される空間量を減少すると共に、オフチップのデータの転送量を減少する。

【0047】図2は、本発明の一実施の形態によるシステムの更に詳細な図を示す。システム200はプロセッサ102とデバッグ回路103とを含み、そのデバッグ回路103はシステムバス105と通信リンクとによって接続され、その通信リンクはクロック信号218、停動信号219、停動済み信号220、トレースデータ221、トレースデータ222、及び状態データ223を含む。

【0048】デバッグ回路103はトレースプロセッサ205を含み、そのトレースプロセッサ205は捕捉バッファ203又はトレースデータラッチ206からの情報と、基準カウンタ217から受け取ったタイムスタンプ情報とを受け取り、FIFO202内に記憶されたトレースデータ内へ受け取ったデータのフォーマットを作る。一実施の形態によれば、FIFO202は、トレースメッセージを記憶するための一時的なエリアとして使われる。FIFO202はトレースメッセージを保持し、トレースメッセージは外部システム106へ送られるのを待ち、システムメモリ内に構成されたトレースバッファに書き込まれるのを待つ。宛先セクタ213は、伝送回路215を介して外部システムに情報を送り、又は、メモリ回路214を介してトレースバッファ（図示せず）内にメモリをセーブ（保存）するようにプログラム可能である。循環トレースバッファとして、又は、トレースバッファ保持モード内に、メモリ回路21

4を構成可能である。循環トレースバッファではトレースバッファの満杯時にトレースメッセージが循環様式で上書きされ、又は、トレースバッファ保持モードではトレースバッファの満杯時にトレース情報が廃棄される。トレース保持モードでは、FIFOのコンテンツは、外部システム106に自動的に出力され、又は、トレースバッファ内に自動的に置かれるのではなく、FIFO内に保持される。例えばデバッグツールのような外部システム106は、FIFOからトレースメッセージを抽出するように構成可能である。

【0049】同様に、トレース情報がプロセッサ又は外部システム106によって読み取られるまで、FIFO202が新しいトレースメッセージを廃棄するように構成可能である。外部システム106はトレース保持モードとして後述される。FIFO202はまた次のように構成可能である。即ち、FIFO202が「頂点」、つまり、最高水準の下でFIFO202内の所定数の地点に到達した時に、新しいトレースが廃棄される（捨て去られる）ように、FIFO202を構成することも可能である。更に、FIFOは循環FIFOモードとして構成可能である。その循環FIFOモードでは、生成された直近のトレースメッセージをFIFOが含むように、FIFO内の旧メッセージが新メッセージによって上書きされる。デバッグ回路が、FIFOトレース保持モード又はトレースバッファ保持モード内にある時、プロセッサ102を停動するために、又は、トレースメッセージを廃棄するためにプログラムされ得る。例えばFIFOトレース保持モードでは、外部システムがFIFOを空にしたり、又は、他の修正措置を取るべきであることを表示するために、トリガ信号が外部システム106に供給される。あらゆるタイプの記憶デバイスが、上述の様々なモードを実行するために、使用可能であることが分かる。

【0050】プロセッサ102をデバッグ回路103に接続するトレースデータバス221は、プロセッサのクロックサイクルの毎に、プロセッサの状態に関する情報を転送可能である。更に、トレースデータバス221は、同一クロックサイクル内において多種多様な状態に監視可能であるように、監視された各プロセッサ状態に関する個別ビットを含むことができる。状態データ224は、プログラムカウンタの現行値と共に捕捉バッファ203にロードされる。トレースプロセッサ205内のトレースメッセージ生成論理は、捕捉バッファ203から状態情報項目を引き出し、対応するトレースメッセージをFIFO202にロードする。

【0051】また、回路103は、状態プロセッサ207、1つ又は複数の事象カウンタ208、1つ又は複数のチェーンラッチ209、性能カウンタ210、及び、トリガラッチ211も含む。プロセッサ102又はバスアナライザ201内に特定の状態が発生した時、状態デ

ータ223、224は次のことを示す。即ち、特定の状態が発生し、状態プロセッサ207が、デバッグ回路103内で1つ又は複数の動作を実行可能であり、例えば、事象カウンタ208をインCREMENT（1つ増や）し、チェーンラッチ209内に記憶された値を設定又は一掃し、性能カウンタ210をインCREMENT（1つ増や）し、トリガラッチ211内に記憶されたトリガアウト信号を制御し、又は、デバッグ回路103内の他の動作を制御するようにしている。状態プロセッサ207とトレースプロセッサ205とは、同一プロセッサとして

【0052】捕捉バッファ203及びトレースデータラッチ206内に記憶されたトレースデータがマルチプレクサ204を介して処理され、トレースプロセッサ205に供給され、プロセッサ102とバスアナライザ201とからの両トレース情報を同一FIFO202内に記憶可能であるようにしている。しかしながら、個別のFIFOも使用可能であることが分かる。

【0053】クロック信号218はプロセッサクロック周波数で実行したプロセッサ102のクロック信号である。信号218は、タイムスタンプカウンタ用のタイマーインCREMENTを決定するプリスケアラ216回路に供給される。例えば、プリスケアラは、プロセッサクロック周波数を整数値によって割り算可能であり、又は、プロセッサクロック周波数値を直接使用することも可能である。プリスケアラ216は、所定時間からの基準カウントを生成する基準カウンタ217にインCREMENT信号を供給する。基準カウンタ217は、タイムスタンプ情報を準備するために、時間の絶対カウントをトレースプロセッサ205に供給する。一実施の形態では、基準カウンタ217はトレースプロセッサ205に40ビットの時間値を供給する。順にトレースプロセッサ205は、生成された最終トレースメッセージからの時差を決定し、トレースメッセージ内のタイムスタンプ情報として時間差を符号化することによりタイムスタンプ情報を圧縮する。

【0054】図3は、トレース情報を処理するためのプロセスを示す。ステップ301でプロセス300が始まる。ステップ302で回路103がプロセッサ102からトレース情報を受け取る。ステップ303で回路103がトレースメッセージを生成し、そしてトレースメッセージを記憶し、又は、ステップ304でトレースメッセージを外部システムに伝送する。更に、回路103は、ステップ305でプロセッサ102から受け取った状態情報を処理し、ステップ306で状態条件を評価して対策を実行する。上述したように、対策は、デバッグ回路103内のレジスタを設定すること、トレースメッセージを生成すること、例外を提起すること、又は、例えばデバッグソフトウェアをロードすると共に実行する

ようにプロセッサ上の対策を実行することを含む。ステップ307でデバッグ回路103が、追加トレース情報を受け取るか否かを決定する。もし受け取る（Yes）ならば、デバッグ回路が捕捉バッファ203又はトレースデータラッチ206内で追加トレースパケットを受け取る。もし受け取らない（No）ならば、追加トレース情報を処理できるまで、回路103が追加トレースデータを捨て去り、又は、プロセッサ102を停動する。ステップ308でプロセス300が終了する。

【0055】図4は、トレース情報を受け取るプロセス400を示す。ステップ401でプロセス400が始まる。ステップ402で回路103は、FIFO202が満杯であるか、即ち図2で説明したような最高水準に到達したかどうかを決定する。もしYesであれば、ステップ408でFIFO202がどんなモードに形成されたかを回路103は決定する。また、回路103は、FIFO202が満杯である表示を外部システム106に供給する。一実施の形態では信号が、外部システム106へ伝送されたトレースメッセージ内でコード化される。その代りに、回路103は、ステップ402でFIFOが最高水準に接近しつつあると決定するかもしれない。ステップ402は、FIFOがその最高水準から記憶場所の制限数内で記憶の1つのレベルに到達したことを決定する。FIFOにおける記憶場所の数は固定され、又は、ソフトウェアを介してユーザによってプログラム可能である。そして、使用された記憶場所の数は、FIFOバッファ内でのエントリ個数をトラックしたレジスタ内に記憶される。FIFO202のレベルが、固定され又はプログラム可能なレベルの値を越えると、プロセッサは停動されるかもしれない。記憶場所の制限数はトレースメッセージ数以上であり、そのトレースメッセージは、例えばプロセッサの停動後のような時間枠内に、プロセッサによって生成可能である。従って、FIFOの満杯前にプロセッサを停動できるので、トレースメッセージを保存できる。FIFO202の満杯を検出する他の方法の使用も可能であることが分かる。

【0056】FIFO202は循環バッファモードに構成可能であり、そこではFIFO202が直近に生成されたトレースメッセージを含むように、FIFO202内の旧トレースメッセージが新トレースメッセージによって上書きされる。循環バッファモード内で生成されたトレースメッセージはアドレスの絶対値と、タイミング情報とを含む。そのタイミング情報は、基準情報の上書き（図11A、11Bと表7とに関して後述される）が圧縮情報の再構成に影響しないように、圧縮されていない。循環バッファモード内では、メモリマップレジスタによってFIFO202のコンテンツが読み込み可能である。ステップ410で回路103がFIFO202内のトレース情報を上書きする。

【0057】もしFIFOが廃棄モードに形成されてい

るならば、トレースメッセージは、FIFO102内に満杯になるまで蓄積される。FIFO202が満杯になると、新しいトレースメッセージが廃棄される。このモード内ではメモリマップレジスタによってFIFO202のコンテンツが読み出し可能になる。

【0058】もしFIFOが停止モードに形成されているならば、FIFO内で空間が利用可能になるまで、捕捉バッファがプロセッサ102を停動させるだろう。トレースメッセージを廃棄しないように、停動モードを使用可能である。

【0059】もしFIFOが満杯でないならば、トレースプロセッサ205が捕捉バッファ203又はトレースデータラッチ206から追加トレース情報を受け取り、ステップ404でトレース情報を処理する。もしステップ412で1つ又は複数のトレースメッセージが失われ、又は、FIFOが上述のような循環バッファモードであり、そして圧縮情報が好まれないならば、ステップ406でトレース情報が非圧縮フォーマットでFIFO202内に記憶可能である。例えば、圧縮トレース情報は、以前に失われたトレースメッセージ内の情報に関する情報を含み、構成不可能になる。従って、絶対情報は新しいトレースメッセージの符号化の際に用いられる。その代りに、ステップ405で、トレースプロセッサは、FIFO202内に情報を記憶する前に、トレース情報を圧縮する。例えば、トレースプロセッサ205は、図11A及び11Bに関して以下に述べるように、圧縮プロトコルに従ってアドレス及びタイムスタンプ値を圧縮可能である。ステップ406で、トレースプロセッサ205はFIFO202内にトレース情報を記憶する。そこでは、直近のトレースメッセージがFIFOの見出しに位置し、最古のトレースメッセージがトレースポートレジスタ212を介して読み出される。

【0060】ステップ407で、デバッグ回路103は、例えば伝送回路215を介してトレースメッセージを外部システムに伝送し、又は、メモリ回路214を介してトレースメッセージを記憶する。一実施の形態では、メモリ回路214がシステムバス105に作動的に結合され、集積回路101と関連したシステムメモリに対して正しい機能を実行する。上述したように、デバッグ回路103は宛先セクタ213を含み、その宛先セクタ213によってユーザがトレースメッセージの宛先を選択可能にする。宛先は、デバッグ回路103内に位置するプログラム可能なレジスタを介して構成可能である。ステップ413で、プロセス400が終わる。

【0061】図5は、トレースバッファモードに基づいてトレース情報を受け取るプロセスのフローチャートを示す。ステップ501でプロセス500が始まる。ステップ502で、トレースバッファが満杯であるか否かをデバッグ回路103が決定する。トレースバッファに関する情報は、エントリの個数とトレースバッファの最大

サイズとに関する情報を含み、デバッグ回路103に記憶される。一実施の形態では、トレースバッファのサイズは、デバッグ回路103内に位置するレジスタを介してプログラム可能である。もしトレースバッファが満杯であるならば、ステップ504でトレースバッファがどのようなモードに形成されたかをデバッグ回路103が決定する。もしトレースバッファが循環バッファモード内で作動中であるならば、ひとたび直近のトレースバッファを含むようにバッファが満杯になると、全てのエントリが新しいエントリによって上書きされるだろう。ステップ506で、回路103がトレースバッファ内のトレース情報を上書きする。

【0062】もしトレースバッファが停動モードに形成されているならば、ステップ505でトレースバッファ内の空間が利用可能になるまで、回路103はプロセッサ102を停動させるだろう。もしトレースバッファが満杯でなければ、回路103は、ステップ503でFIFO内の最終メッセージをトレースバッファへ転送する。説明したように、メモリ回路214はシステムバス105にアクセスして場所及びシステムメモリ内にトレース情報を書き込み可能である。

【0063】上述したように、状態値はデバッグ回路103へ転送可能である。その状態値は、プロセッサ102の1つ又は複数の状態を表示する1つ又は複数の信号によって表される。例えば、ウォッチポイントチャンネルはプロセッサ102内で定義可能である。そのプロセッサ102は、コンピュータのメモリ内でアクセスされたデータアドレスのような特別なデータ値、システム105上に位置されたモジュールのアドレス、プロセッサによって実行されたオペランドのアドレス、又は、1つ又は複数の所定値によってマッチ（一致）可能であるプロセッサ内の他の任意な状態を有する。ウォッチポイントチャンネルはマッチングメカニズムを含み、それによってプロセッサ102内のレジスタに書き込まれたデータ値がプロセッサ102内のデータ値と比較される。そのプロセッサ102は、命令アドレス、命令値、オペランドアドレス、性能カウンタ、事象カウンタなどを含む。

【0064】ウォッチポイントチャンネルに関連したコントローラが、マッチされた時に、通信リンク215を介してデバッグ回路103に信号を供給可能である。この信号は、デバッグ回路103に連絡されたプロセッサ102内の特定ウォッチポイントチャンネルの状態を表示する状態ビットの形をとる。また、デバッグ回路103によって異なるデバッグ操作を実行するために、ウォッチポイントチャンネルに対応する状態ビット値は組合わせ可能であり、これらの状態ビット値は交信も可能である。

【0065】同様の方法では、デバッグ回路103は、デバッグ操作内での使用のために、多数の信号をプロセ

ッサへ供給する。特に、デバッグ回路103は多数のビット値を供給可能にする。その多数のビット値は、プロセッサ102内での特定事象をトリガすることへの必須条件として作動する。それから、これらの事象は、デバッグ回路103によって受け取られるように、トレース情報又はその他の状態情報を生成する。

【0066】更に、ウォッチポイントチャンネルによってプロセッサ102がトレースパケットを生成させられ、場合によっては、例外を生成させる。ウォッチポイントチャンネル自体もまた、そのチャンネルが状態情報を生成するか否かを決定する必須条件と、特定のウォッチポイントに関してマッチングが起きる否かを表示するマッチ条件と、ウォッチポイントチャンネルのマッチングに基づいて対策が取られるならば、それはどんなタイプの対策であるのかを決定する対策条件とを有している。

【0067】既に述べたように、多数のウォッチポイントはプロセッサ102とデバッグ回路103との中で形成可能である。これらのウォッチポイントは、プロセッサ102又はデバッグ回路103のどちらかに位置されたデータラッチ内に記憶されていた状態値を決定する。1つのデータラッチの出力は、別のラッチ（ウォッチポイントが同時に「チェイン」されている）への入力として役立ち、又は、ウォッチポイントチャンネル用の必須状態として機能する。ウォッチポイント及びデータラッチのこれら及び他の特徴は同時係属中の米国特許出願で更に詳細に記述されている。その米国特許出願は、D. Edwards等によって1999年10月1日に出願（弁護士事件整理番号（Attorney Docket Number）99-TK-263）され、「MICROCOMPUTER DEBUG ARCHITECTURE AND METHOD」（マイクロコンピュータデバッグアーキテクチャ及び方法）と名付けられ、その全体を参照することによって本明細書に結合されている。

【0068】図6は、異なるトレースモードに基づいてトレースメッセージを生成するためのプロセス600を示す。ステップ601ではプロセス600が始まる。ステップ601で、回路103は、どのトレースモードが作動可能であるかを決定する。回路103は、マルチトレースモードと呼ばれている物で構成され、それにより受け取った各ウォッチポイント状態表示のためにトレースメッセージが生成される。それとは逆に、回路103を単一トレースモードで形成することも可能であり、それによって一連の類似ウォッチポイント状態のためにシングル（単一）トレースメッセージが生成されるだろう。例えば、生成されたものから、別の即値トレースメッセージの生成を避けるために、単一トレースモードが使用可能である。例えば、命令アドレスウォッチポイント場所をアドレスレンジA(n)からA(m)上で考察する。レンジA(n)からA(m)の間の第1命令

実行は命令アドレストレースメッセージを生成し、レンジA(n)からA(m)内における全ての連続実行もトレースメッセージを生成するだろう。多くの状況では、第1トレースメッセージだけが重要である。そして、その実行がレンジの外側に進行するまで、即ち、ウォッチポイントチャンネルがマッチングできなくなるまで、レンジA(n)からA(m)内の連続メッセージは重要ではない。従って、トレースは例外ハンドラを使用せずに収集される。

10 【0069】ステップ605で、回路103は、ウォッチポイントの「ヒット」又は状態マッチを示す状態変化を受け取る。ステップ606で、トレースプロセッサ205がトレースメッセージを生成する。それから、回路103はステップ607で別のウォッチポイントヒットを受け取る。ステップ608で、回路103は、ウォッチポイントがマッチしたか否かを決定する。もしそうでない(No)ならば、回路103は、追加トレースメッセージを生成せずに追加ウォッチポイントを受け取る。もしウォッチポイントがマッチしない(Yes)ならば、回路103は追加トレースメッセージを生成する。

20 【0070】図7は、一般的なトレースメッセージ701のフォーマットを示す。本発明の一実施の形態では、一般的なトレースメッセージフォーマットが定義される。それによって、トレースメッセージフィールドが、異なるタイプのトレースメッセージに提供され、メッセージ内に提供されたトレース情報が、トレースメッセージの分析を実行するために必要な全ての情報を含む。トレースメッセージ701は、トレースメッセージのタイプを示すメッセージタイプのフィールド702を含み、

30 トレースメッセージに必要な対策を決定するために外部システム106のソフトウェアツールによって使用可能である。例えば、即値対策を必要としないが、ソフトウェアツールのメモリ内に記憶された背景トレースメッセージが当該トレースメッセージあることを、メッセージタイプが示すならば、そのメッセージは単に記憶されるだろう。更に、トリガタイプのトレースメッセージは、当該トレースメッセージ上の即値対策が必要とされることをソフトウェアツールに示す。

40 【0071】ソースモジュール703は、トレースメッセージ内の情報を提供するオンチップソースモジュールを識別する。例えば、ソースモジュール703はプロセッサ102内のウォッチポイントコントローラ、バスアナライザ201、又は、システムバス105を介して位置された他のデバイスを識別する。事象タイプ704は、トレースメッセージを生成するソースモジュール内のウォッチポイントチャンネルを形成する。プロセッサ102が停動モード内であれば、FIFO202内に利用可能な空間が存在しないために、生成されるべき現在のトレースメッセージより前のかかなり長い間に停動された否かを、過剰停動705が示す。非停止モードでは、

FIFO202内で利用可能な空間が存在しないために、現在のトレースメッセージ前に1つ又は複数のトレースメッセージが捨て去られたことを示すように、このビットが設定される。

【0072】PCアブソリュート706は、PCフィールド710が絶対アドレス又は相対アドレスを含むか否かを定義する。相対アドレスは、例えば、前のトレースメッセージ内に送られた直近プログラムカウンタ値からの符号付きオフセットである。ステータス707は、各ウォッチポイントチャンネルタイプに関して特有な状態である。もし回路103がトレースメッセージ内にタイムスタンプ情報を含むように形成されているならば、タイムスタンプ708はオプションのフィールドである。タイムスタンプ708は、例えば、最終基準トレースメッセージ（基準トレースメッセージは更に詳細に後述される）が生成されたので、多数のタイマーインクリメントを指定する値である。ASID709は、プロセス識別子情報を含むオプションのフィールドである。ASIDフィールド709は、例えばトレースバケット内でトレース情報を生成したソフトウェアプロセスのプロセス*20

* 識別子を報告するために、プロセス識別子を含む。また、もしASID値がデバッグプロセスにとって重要でないならば、又は、もしASIDが特定値であることが理解されるならば、ASIDフィールド709情報は省略可能である。例えば、ASIDフィールド709の不必要なウォッチポイントチャンネルによって生成されたトレースメッセージを、ASIDの任意の特定値上でマッチするように、ウォッチポイントチャンネルが設定されているならば、ウォッチポイントから生成されたトレースメッセージの単なる存在がASIDの値を示す。PC710は、プロセッサ102のプログラムカウンタ、即ち、相対アドレスのような圧縮アドレスの絶対値を含む可変レングスフィールドである。相対アドレスは、例えば、前のトレースメッセージ内で送られた直近プログラムカウンタ値からの符号付きオフセットである。

【0073】以下の表1は、一般的なトレースメッセージの典型的なフォーマットを示す。

【0074】

【表1-1】

| 【表1-1】 一般的なトレースメッセージフィールド | | | |
|------------------------------|------|----------|---------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 (DTEC 背景トレースメッセージ) のデバッグメッセージフィールド値の内容を定義し、0b011 (DTEC トリガトレースメッセージ) はトレースメッセージのタイプを表示する |
| ソースモジュール | 3ビット | [3, 5] | トレースメッセージ内情報を提供するオンチップソースモジュールを定義する。 値-記述 0 - プロセッサウォッチポイントコントローラ 1 - バスアナライザ 2 - 追加プロセッサ又は他のモジュール内のウォッチポイント ジック用のリザーブ |
| 事象タイプ | 5ビット | [6, 10] | トレースメッセージを生成するソースモジュール内のウォッチポイントチャンネルを定義する。 値がプロセッサ及びバスアナライザ内のウォッチポイントチャンネルを識別する。 |

表1：一般的なトレースメッセージフィールドの例

【0075】

【表1-2】

【表1-2】

| 一般的なトレースメッセージフィールド | | | |
|--------------------|------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| 過剰停動 | 1ビット | [11] | このビットは、停動モードレジスタが停動モードか否かに依存して二つの意味を持つ。 停動モードでは、デバッグモジュール P1P0 内に利用可能な空間がないから、生成されるこのトレースメッセージ前の不定時間の間にプロセッサ 102 が停動された時、このビットが決定される。 非停動モードでは、デバッグモジュール P1P0 内に利用可能な空間がないから、1つ又はそれ以上のトレースメッセージが1つ前に廃棄されたことを示すように、このビットが決定される。 |
| PC 絶対 | 1ビット | [12] | PC フィールドが4バイト絶対アドレス、又は、1又は2バイトの相対アドレスを含むか否かを定義する。相対アドレスは、前のトレースメッセージ (任意タイプ) で送られた直近の PC 値からの符号付きオフセットである。 値-1555 0-相対アドレスオフセット 1-絶対4バイトアドレス |
| その他 | 4ビット | [13, 15] | 各ウォッチポイントチャンネル用のタイプに特定。 |
| タイムスタンプ | 1バイト | N/A | W P チャンネルの対策が enable_trace_timestamp=1 を含む時に、この任意フィールドがトレースメッセージ内に発生する。 基礎トレースメッセージが生成されたので、この1バイト値がタイムインクリメント値を特定する (表7参照)。 |

表1：一般的なトレースメッセージフィールドの例

【0076】

30 【表1-3】

【表1-3】

| 一般的なトレースメッセージフィールド | | | |
|--------------------|------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| ASID | 1バイト | N/A | 任意 ASID にマッチするような原因ならば、ウォッチポイントチャンネル毎に、この任意フィールドが発生する。 ウォッチポイントチャンネルの稼働状態が asid_enable=1 を含む時、ASID フィールドはトレースメッセージ内に現れない。 |
| PC | 1, 2又は4バイト | N/A | もし PC 絶対値が「0」ならば、このフィールドは、前のトレースメッセージ (任意タイプ) 内で送られた直近 PC 値からの符号付きオフセットとして1又は2バイトの相対アドレスである。 もし PC 絶対値が「1」ならば、このフィールドはシャドウ PC の4バイト絶対値から成る。 |

表1：一般的なトレースメッセージフィールドの例

【0077】あるタイプのトレースメッセージに特有なトレースメッセージが生成可能である。例えば、固有なトレースメッセージが生成されると、ウォッチポイントチャンネルを引き起こした状態のように、プロセッサ1

02内で特定の事件(出来事)を示す。以下の表2～表7は、様々なウォッチポイント用のトレースメッセージタイプにおける実施の形態を示す。

【0078】

【表2-1】

【表2-1】

| 命令アドレスウォッチポイントトレースメッセージ (最小3バイト、最大8バイト) | | | |
|-----------------------------------------|------|----------|--------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 又は 0b011 |
| ソースモジュール | 3ビット | [3, 5] | 0 (ウォッチポイントコントローラ) |
| 事象タイプ | 5ビット | [6, 10] | 0x00 から 0x03 |
| 過剰停動 | 1ビット | [11] | |
| PC絶対 | 1ビット | [12] | |
| 予約済み | 3ビット | [13, 16] | 使用されず |

表2: 命令アドレスのウォッチポイントトレースメッセージ

【0079】

* * 【表2-2】

【表2-2】

| 命令アドレスウォッチポイントトレースメッセージ (最小3バイト、最大8バイト) | | | |
|-----------------------------------------|---------------|----------|----|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| タイムスタンプ | 0 又は 1 バイト | N/A | |
| ASID | 0 又は 1 バイト | N/A | |
| PC | 1, 2 又は 4 バイト | N/A | |

表2: 命令アドレスのウォッチポイントトレースメッセージ

【0080】

※20※ 【表3-1】

【表3-1】

| オペランドアドレスウォッチポイントトレースメッセージ (最小11バイト、最大16バイト) | | | |
|----------------------------------------------|------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 又は 0b011 |
| ソースモジュール | 3ビット | [3, 6] | 0 (ウォッチポイントコントローラ) |
| 事象タイプ | 5ビット | [6, 10] | 0x04 から 0x05 |
| 過剰停動 | 1ビット | [11] | |
| PC絶対 | 1ビット | [12] | |
| 予約済み | 1ビット | [13] | |
| データフィールドサイズ | 2ビット | [14, 15] | このトレースメッセージのデータフィールドを生成したメモリ書き込みオペレーションのサイズを定義する。 <u>値-記述</u> 0b00-1 バイト。ウォッチポイントをヒットした命令が記憶場所への1バイト書き込みを実行した。 0b01-2 バイト (同上) 0b10-4 バイト (同上) 0b11-8 バイト (同上) |

表3: オペランドアドレスのウォッチポイントトレースメッセージ

【0081】

【表3-2】

【表3-2】

| オペランドアドレスウォッチポイントトレースメッセージ (最小11バイト、最大16バイト) | | | |
|----------------------------------------------|------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| タイムスタンプ | 0又は1バイト | N/A | |
| ASID | 0又は1バイト | N/A | |
| PC | 1, 2又は4バイト | N/A | |
| アドレスデータ | 8バイト | N/A | <p>このフィールドは、メモリ書き込みオペレーションに関連したデータの8バイト、又は、データの4バイトを加えて4バイトオペランドアドレスの何れかを含み、オペランドアドレス内のデータフィールドの値に依存する。</p> <p>例えば、もしオペランドアドレストレースデータフィールドの値==0であれば、このトレースメッセージフィールドはデータの8バイトを含み、Data_field_sizeフィールドは、実際にどのバイトがメモリに書き込まれたかを定義する。</p> <p>もしオペランドアドレスデータフィールドの値==1であれば、このトレースメッセージフィールドは、ビット位置[31, 0]内のデータの最下位4バイトを加えてビット位置[63, 32]内のデータの絶対4バイトオペランドアドレスを含む。Data_field_sizeフィールドは、実際にメモリに書き込まれたバイト数を定義する。</p> <p>最下位4ビットだけがトレース情報内にふくまれたとしても、それは、Data_field_size=3 (8バイト) のために可能であることに注意しなさい。</p> |

表3：オペランドアドレスのウォッチポイントトレースメッセージ

【0082】

【表4】

【表4】

| 命令値ウォッチポイントトレースメッセージ (最小3バイト、最大16バイト) | | | |
|---------------------------------------|---------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 又は 0b011 |
| ソースモジュール | 3ビット | [3, 5] | 0 (ウォッチポイントコントローラ) |
| 事象タイプ | 5ビット | [6, 10] | 0x06 から 0x07 |
| 過剰停動 | 1ビット | [11] | |
| PC 絶対 | 1ビット | [12] | |
| データフィールドサイズ | 3ビット | [13, 15] | このトレースメッセージの宛先データフィールドのサイズを定義する。 値-記述 0b0xx-ウォッチポイントをヒットした命令が記憶場所に書き込まなかったために含まれるデータフィールド。 0b100-1バイト。ウォッチポイントをヒットした命令が記憶場所へ1バイト書き込みを行った。 0b101-2バイト記入 (同上) 0b110-4バイト記入 (同上) 0b111-8バイト記入 (同上) |
| タイムスタンプ | 0 又は 1 バイト | N/A | |
| ASID | 0 又は 1 バイト | N/A | |
| PC | 1, 2 又は 4 バイト | N/A | |
| 宛先データ | 0, 1, 2, 4 又は 8 バイト | N/A | 記憶場所に書き込んだ命令によって生成されたトレースメッセージ内に存在する。このフィールドのサイズはデータフィールドサイズ値によって決定される。 |

表4：命令値のウォッチポイントトレースメッセージ

【0083】

【表5】

【表5】

| 分岐ウォッチポイントトレースメッセージ (最小4バイト、最大12バイト) | | | |
|--------------------------------------|------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 (普通トレースメッセージとして定義される) |
| ソースモジュール | 3ビット | [3, 5] | 0 (ウォッチポイントコントローラ) |
| 事象タイプ | 5ビット | [6, 10] | 0x08 |
| 過剰停動 | 1ビット | [11] | |
| SBC 絶対 | 1ビット | [12] | |
| DEST 絶対 | 1ビット | [13] | |
| 予約済み | 2ビット | [14, 15] | |
| タイムスタンプ | 0又は1バイト | N/A | |
| ASID | 0又は1バイト | N/A | |
| ソースアドレス | 1, 2又は4バイト | N/A | もし SBC 絶対が「0」であれば、このフィールドは、前のトレースメッセージ (任意タイプ) 内に送られた直近 PC 値からの符号付きオフセットとしての1又は2バイトの正確アドレスである。 もし SBC 絶対が「1」であれば、このフィールドはシャドープCの4バイト絶対値から成る。 |
| Destn アドレス | 1, 2又は4バイト | N/A | もし DEST 絶対が「0」であれば、このフィールドは、前のトレースメッセージ (任意タイプ) 内に送られた直近 PC 値からの符号付きオフセットとしての1又は2バイトの正確アドレスである。 もし DEST 絶対が「1」であれば、このフィールドは分岐用の宛先アドレスの4バイト絶対値から成る。 |

表5：分岐のウォッチポイントトレースメッセージ

【0084】

* * 【表6】

【表6】

| 高速プリントウォッチポイントトレースメッセージ (15バイト) | | | |
|---------------------------------|------|----------|-----------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b011 (トリガトレースメッセージとして定義される) |
| ソースモジュール | 3ビット | [3, 5] | 0 (ウォッチポイントコントローラ) |
| 事象タイプ | 5ビット | [6, 10] | 0x08 |
| 過剰停動 | 1ビット | [11] | |
| 予約済み | 4ビット | [12, 15] | |
| ASID | 1バイト | N/A | ASID がトレースメッセージ内に含まれる。 |
| PC | 4バイト | N/A | 過剰の PC アドレスが常に送られる。 |
| FPF データ | 8バイト | N/A | 高速プリントレジスタに書き込まれたデータは常に64ビット位である。 |

表6：高速プリントのウォッチポイントトレースメッセージ

【0085】

【表7-1】

【表7-1】

| バスアナライザウォッチポイントトレースメッセージ (最小7バイト、最大20バイト) | | | |
|-------------------------------------------|------|----------|--------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b010 又は 0b011 |
| ソースモジュール | 3ビット | [3, 5] | 1 (バスアナライザ) |
| 事象タイプ | 5ビット | [6, 10] | 0x00 から 0x01 |
| 過剰停動 | 1ビット | [11] | デバッグモジュール FIFO 内に利用可能な空間がなかったため、1つ又はそれ以上のトレースメッセージがその1つ前に廃棄されたことを表示するように設定される。 |

表7: バスアナライザのウォッチポイントトレースメッセージ

【0086】

* * 【表7-2】

【表7-2】

| バスアナライザウォッチポイントトレースメッセージ (最小7バイト、最大20バイト) | | | |
|-------------------------------------------|--------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位置 | 記述 |
| マッチロス | 1ビット | [12] | 以下の何れかの時にヒットが起きたので、捕捉されるはずのバスセル/トークンが失われたことを表示するように設定する: 以前又は最新の捕捉セル/トークンがデバッグモジュールに転送されつつあり、又は、 捕捉バッファが凍結される。 ウォッチポイントヒットがこれらの追加バストランザクション用に登録されない。 |
| ADDR 絶対 | 1ビット | [13] | |
| 予約読み | 2ビット | [14, 15] | |
| タイムスタンプ | 0 又は 1 バイト | N/A | |
| ソース | 1バイト | N/A | |
| 宛先 | 1バイト | N/A | |
| 演算コード | 1バイト | N/A | |
| トランザクション ID | 1バイト | N/A | |
| データマスク | 0 又は 1 バイト | N/A | |
| アドレス | 1, 2 又は 4バイト | N/A | もし ADDR 絶対が「0」であれば、このフィールドは、このウォッチポイントに対して前のトレースメッセージ用に算定されたバストランザクションアドレスから符号付きオフセットとしての1又は2バイトの圧縮アドレスである。 もし ADDR 絶対が「1」であれば、このフィールドはトランザクションアドレスの4バイト絶対値から成る。 |
| トランザクションデータ | 0 又は 8 バイト | N/A | もし捕捉されたバスパケット (リクエスト又はレスポンス) がデータを含むならば、このフィールドが含まれる。トレースメッセージがデータを含むか否かを決定するために演算コードが用いられる。 |

表7: バスアナライザのウォッチポイントトレースメッセージ

【0087】図8はトレースバッファ227の一実施の形態を示す。既に検討したように、FIFO202は、固定サイズ又は変更自在な長さである1つ又は複数のトレースメッセージ801を受け取り、トレースバッファ227にそのトレースメッセージを供給する。デバッグ回路103は、トレースバッファ227を維持するために、多数のレジスタ804～807を含む。例えば、回

路103は、次のトレースメッセージを書き込む場所をトレースバッファ内で決定するために、レジスタ804を含む。このエリアは、バッファ内の第1エン트리802に対応する。また、回路103は、トレースバッファ227の終端部にポインタを保持するレジスタ805を含み、それによってポインタがバッファ内の最終エン트리803を指す。回路103は、バッファ内のエン트리

数のトラック（軌跡）を維持するレジスタ806と、トレースバッファモードを含むレジスタ807とを有する。既に述べたように、図5を参照して、トレースバッファは、トレース情報を上書きし、又は、追加入力トレースメッセージ801を廃棄するように、構成可能である。既に検討したように、トレースバッファ227は、固定長のトレースメッセージA～C（項目808～810）、又は、トレースメッセージD～F（項目811、812A、812B、813）のように可変長なトレースメッセージを含む。トレースメッセージがトレースバッファ227の終端部から読み出し可能であり、又は、

トレースエントリが位置決めされ、トレースバッファ227のコンテンツをマッピングしたインデックスのテーブル索引を介すように、トレースバッファ227から読み出し可能である。

【0088】図9はトレースバッファの一実施の形態を示す。トレースバッファ901は可変長又は固定長どちらかのメッセージを含み、最大サイズのメッセージエントリ902は3*64ビットのワード内に適合する。一形態では、デバッグ回路103は、固定した3*64ビットの間隔でトレースメッセージをトレースバッファ901内に読み込む。しかしながら、トレース情報は任意のフォーマットで、且つ任意のインターバル数で記憶可能であることが分かる。ポインタ903は、次のエントリがバッファ901内に書き込みできるように、現在の書き込み場所を指示する。トレースメッセージの抽出時に、デバッグ回路103は、トレース情報の順序が決定されるように、トレースバッファのモードに気づくべきである。更に、そのモードは、アドレス情報が圧縮されたか否かを決定する。

【0089】回路103はトレースポートレジスタ212を含み、回路103によってデバッグソフトウェアがFIFO202からトレースデータを読み出すことを可能にする。例えば、FIFOがトレース保持モード又は循環モードであるとき、情報はFIFO202から読み出し可能になる。トレースデータは、一回で1つのトレースメッセージを抽出可能であり、三つのレジスタはトレースポートレジスタ212として使用され、例えば3*64ビット以下であるトレースエントリのうち最大サイズを読み出す。3個のトレースポートレジスタ212は間接的にFIFOに結合される。FIFO202からトレースポートレジスタ212に最古のトレースメッセージを転送するための1つのメカニズムは次の順序を含む。

【0090】(A) デバッグソフトウェアはリクエストを供給してFIFO202からトレースポートレジスタ212への最古のトレースメッセージの転送を開始する。

【0091】(B) デバッグソフトウェアは、レジスタ値が1になるまで肯定応答レジスタを読み出し、転送が完了したことを示す。例えば、この肯定応答状態は、デバ

ッグソフトウェアによって次の転送リクエストまで存在する。

【0092】(C) そこで、トレースポートレジスタ212からトレースデータを読み出すことができる。例えば、下位順序レジスタが最初に読み出されてトレースメッセージのサイズを決定するが、レジスタは任意の順序で読み出し可能であり、且つ何回でも読み出し可能である。

【0093】(D) 別のトレースメッセージ転送をリクエストする前に、デバッグソフトウェアは、FIFO202内にもっと多くのトレースメッセージが存在するかどうかを決定する。

【0094】既に検討したように、デバッグ回路103は、プロセッサ102から制御信号を供給及び受信する捕捉バッファ203を含む。捕捉バッファ203は、バッファ1002の状態を監視するバッファ制御回路1001を含む。バッファ1002はプロセッサ102からトレースデータを受け取る。もしバッファ1002が満杯であり、又は、所定個数内のエントリが満杯であるならば、バッファ制御回路1001が、トレースメッセージのロス（喪失）を防止するためプロセッサの実行パイプラインを停動するようにプロセッサ102に指示する。特に、バッファ制御回路は、実行パイプラインを停動するため停動信号をプロセッサ102に及ぼす。ある状況では、プロセッサ102が、停動信号を受け取った下で即座に実行パイプラインを停動せず、又は、制限されたプロセッサのクロックサイクル数内で実行パイプラインを空にする。この状況の何れかで、プロセッサ102は、トレースデータを生成して更にデバッグ回路103へ伝送する。次のようにバッファ1002を監視するために、バッファ制御回路は構成可能である。即ち、バッファ内のエントリ数が、満杯（フル）状態へのバッファの接近を示す最高水準を超えた時、プロセッサ102が、自身の完全な停動前に生成したトレースデータに何のロスなく停動される。また、バッファ制御回路1001は、FIFO202の状態に基づかれたプロセッサ102を停動することも可能である。例えば、トレースプロセッサ205又は捕捉バッファ203はFIFO202の満杯を監視し、そして同様な方法で、プロセッサ102を停動するために停動信号を供給することができる。

【0095】図11A及び11Bは、本発明の一実施の形態に従った圧縮済みデータフォーマットを示す。外部リンクを介して送られたトレースメッセージのサイズを最小化し、トレースメッセージの記憶容量を最大化するために、トレースメッセージは圧縮される。例えば、プログラムカウンタのアドレスとバスアナライザのアドレスとは圧縮可能である。符号化方法が用いられ、それにより1バイト又は2バイトを使って7ビット又は14ビットのいずれかの符号付きアドレスオフセットを表す。

これらのオフセットは同じタイプにおける以前のアドレスに関係している。もし7ビット又は14ビットのオフセット値としてアドレスを表すことができないならば、その代わりに、絶対32ビット値を符号化可能である。オフセット値の関係するトレースメッセージに先行する基準内の情報がロスされ、又は、別の理由で利用不可能であるならば、オフセット値は選ばれないだろう。オフセット値が選ばれない状況では、絶対値情報が提供される。例えば、トレース保持又は循環モード内でFIFO 202が作動している時に、基準情報が失われ、上書きされ、又は、別の理由で利用できない可能性が存在する。更に、オフセット値を基準とする絶対値データを含む基準メッセージが供給可能である。例えば、オフセット値がトレースメッセージから取り去れていると、デバッグ回路103はオフセット値に関する基準として使用可能な絶対値を含む基準メッセージを生成可能であり、アドレスが絶対値として符号化される。

【0096】本発明の一形態によれば、プログラムカウンタのアドレスは、32ビット絶対アドレス、又は、以前のプログラムカウンタのアドレスに関する7ビット又は14ビット値の何れかに符号化される。その以前のプログラムカウンタのアドレスは以前のトレースメッセ

ージ内で符号化される。バスアナライザのアドレスは、32ビット絶対アドレス、又は、以前のバスアナライザのアドレスに関する7ビット又は14ビット値の何れかに符号化される。その以前のバスアナライザのアドレスは以前のトレースメッセージ内で符号化される。図11Aに示すように、アドレスオフセットは、新しいアドレスから以前のアドレスを差し引くことによって算定可能である。図11Bに示すように、アドレスは符号付きアドレスオフセットでも可能であるので、符号付きビットSは正のアドレスオフセット用に「0」の値を有している。図11Bに示うように、第1バイトのビット7は、第2バイトが後続するか否かを示すために用いられる。アドレスの絶対又は相対符号化は、一般的なトレースメッセージ701のPC絶対706フィールドによって示される。デバッグソフトウェアが、トレースメッセージ情報を分析している時に、アドレスと次のトレースメッセージとを再構築するための基準として絶対アドレスを使用する。

【0097】例えば、図11A及び図11Bで示されたように、本発明の一実施の形態に従った圧縮デコードルーチンの「C」プログラミング言語の実行が以下に与えられる。即ち、

```

/ _DecodeCompressedOffset
pre: byteStream points to signed compressed value
post: returns decoded value
/
int _DecodeCompressedOffset(char *byteStream)
{
    int result;
    if((byteStream[0] & 128)==0){
        // the value is a 1 byte value
        //
        result=(byteStream[0] & 0x3F); // extract the least significant 6 bits
        // check if it needs inverting
        if(byteStream[0] & 64){
            result=-result;
        }
    } else {
        // its a 2 byte value
        //
        // extract the least significant 7 bits
        result=(byteStream[0] & 0x7F);
        // additionally, extract the most significant 7 bits
        result=result|((byteStream[1] & 0x7F)<<7);
        // check if it needs inverting
        if (byteStream[1] & 0x40){
            result=-result;
        }
    }
    return result;
}

```

である。

【0098】他の圧縮アルゴリズムを使ってプログラムカウンタと、アドレス情報と、トレース情報内に含まれた他のパラメータとを圧縮することも可能であることが分かる。

【0099】既に検討したように、デバッグ回路103は、タイムスタンプ情報を提供する基準カウンタ217を含む。本発明の別の実施の形態では、トレースメッセージは、自身のタイミング情報を表すタイムスタンプを含む。プロセッサ102とバスアナライザ201との両方によって生成されたトレース情報にタイミング情報を加えることが可能である。1つの単一基準カウンタ217を用いることにより、両タイプのトレースは1つの単一時間領域内で互いに密着可能である。既に検討したように、デバッグ回路103はブリスケーラ216を含み、そのブリスケーラ216によってデバッグユーザが、デバッグされているアプリケーションに適するように、タイムスタンプインクリメントのサイズを設定可能である。

【0100】本発明の一形態によれば、図11A及び図11Bに関して既に検討したように、情報がシステム106で最新に維持されるように、基準メッセージが所定間隔内で外部システム106に送られる。特に、基準メッセージは、システム106がデバッグ回路103内で*

* 現在時間を通知するように、外部システム106へのタイミング情報を示す。また、基準メッセージに従ったトレースメッセージ内の相対タイミング情報は、基準メッセージのタイミング情報から算定可能である。更に、基準メッセージに続くトレースメッセージ内の相対プログラムカウンタの情報が、基準メッセージプログラムカウンタ値から算定可能であるように、基準メッセージはデバッグ回路103内に記憶されたプログラムカウンタ値のコピーを含む。その上、アドレス情報もまた、オフセットアドレスを算定するために基準メッセージ内に含むことも可能である。タイミング情報のプログラムカウンタとアドレス情報とは絶対値でもよい。

【0101】また、基準メッセージは、所定個数の時間間隔（タイムインターバル）よりも大きい連続休止期間中に、外部システム106に送られる。例えば、所定個数の時間間隔は256回の間隔でもよい。即ち、もし伝送回路215がトレースメッセージの送信に先立って256回の間隔より長い間休止していたならば、基準メッセージは、次のトレースメッセージの前に、FIFO202内に挿入可能である。

【0102】以下の表8は、基準メッセージのフォーマットとコンテンツとの一実施の形態を示す。

【0103】

【表8】

| 基準メッセージ (14バイト) | | | |
|-----------------|------|--------------|--------------------------------------------------------------------------------------------|
| フィールド | サイズ | 見出しビット位 置 | 記述 |
| メッセージタイプ | 3ビット | [0, 2] | 0b100 |
| 予約済み | 5ビット | [3, 7] | |
| タイム値 | 5バイト | N/A | デバッグモジュール内の40ビットタイムスタンプカウンタの値。 |
| PC アドレス | 4バイト | | このメッセージの時間でシャドウプログラムカウンタの値が生成される。このアドレスは新しい基準PC値になり、後述するバスアナライザトレースメッセージ内の相対アドレスはこの値に基づく。 |
| BA アドレス | 4バイト | | 超高速バスアナライザに接続される2対4バイト基準アドレス。この値は新しいバスアナライザ基準アドレスになり、後述するバスアナライザトレースメッセージ内の相対アドレスはこの値に基づく。 |

表8：基準メッセージ

【0104】上述したように、デバッグ回路103は、タイムスタンプがトレースメッセージ内に含まれるか否かを決定するレジスタを含む。もし含まれるならば、図7のタイムスタンプフィールド708は、最終基準メッ

セージからの時間差を指定するために用いられる。その代りに、タイムスタンプフィールド708がタイムスタンプの絶対値を含むこともあり得る。

【0105】本発明の様々な実施形態について上述した

が、これらの実施の形態は例として示されたに過ぎず、限定的意味を持つものでないことが分かる。従って、本発明の広がり及び範囲は前述の典型的な実施の形態のいずれによっても限定されるものでなく、次の請求の範囲とそれらの等価のものに従ってのみ定義される。

【0106】

【発明の効果】本発明によれば、トレース情報を直接的にどのように処理し、及び、プロセッサオペレーションでどのように割り込むかを明確に述べた割込なし方法が提供される。詳細には、トレースシステムはデバイスを含み、そのデバイスはプロセッサから分離して作動し、プロセッサの内部クロック速度で作動し、又は、トレース情報のロスがプロセッサオペレーションに影響しないようなモードで作動する。更に、収集されたトレース情報は、トレース動作を実行するのに要する全ての情報を含み、プロセッサは、例えばデバッグツール上で実行したソフトウェアプログラムによって追加情報を入手するために、割り込む必要がない。

【0107】そして、レース情報はトレースシステムによって圧縮される。情報を圧縮することによって、トレース情報は、低帯域幅リンクを介して伝送するために保存され、オンチップトレース記憶を最大化する。例えば、トレース情報は、タイムスタンプ及びアドレス情報を圧縮することによって圧縮可能である。更に、トレース情報は、例えば特定のオペレーションタイプにおける1つのトレースバケットのように、重複タイプの情報を省略することによって圧縮可能である。また、情報は、トレース情報を生成する規準を予め決定することによってフィルタ可能である。情報をフィルタし、重複の情報オンチップを除去することにより、外部システムへのリンクの帯域幅要件と、オンチップ記憶要件とが軽減される。トレースメッセージは、メッセージ内のデータフィールドの絶対値を相対値に取り替えることによって、圧縮可能である。圧縮情報の絶対値を含んだ周期的な基準メッセージは、相対値に基準を提供するように設けられる。

【図面の簡単な説明】

【図1】本発明の一実施の形態に従った集積回路のブロック図である。

【図2】本発明の一実施の形態に従った集積回路の詳細なブロック図である。

【図3】トレース情報を処理するためのプロセスのフローチャートである。

【図4】トレース情報を受け取るためのプロセスのフローチャートである。

【図5】トレースバッファモードに基づいたトレース情報を受け取るためのプロセスのフローチャートである。

【図6】異なったトレースモード内でトレースを生成するためのプロセスのフローチャートである。

【図7】トレースメッセージのブロック図である。

【図8】本発明の一実施の形態によるトレースバッファのブロック図である。

【図9】トレースバッファの一実施の形態を示すブロック図である。

【図10】捕捉バッファの詳細な図である。

【図11A】本発明の一実施形の態に従った圧縮データフォーマットを示す図である。

【図11B】本発明の一実施の形態に従った圧縮データフォーマットを示す図である。

【符号の説明】

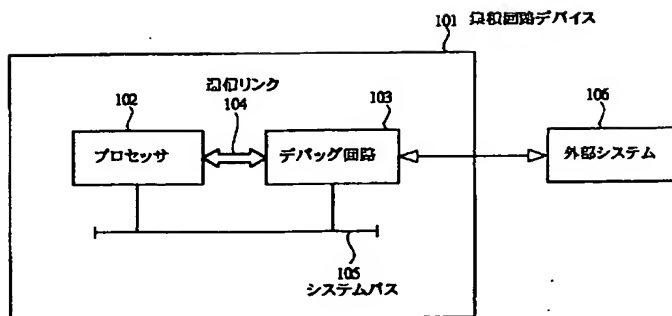
| | |
|-----|-------------|
| 101 | 集積回路デバイス |
| 102 | プロセッサ |
| 103 | デバッグ回路 |
| 104 | 通信リンク |
| 105 | システムバス |
| 106 | 外部システム |
| 201 | バスアナライザ |
| 202 | FIFO |
| 203 | 捕捉バッファ |
| 204 | マルチプレクサ |
| 205 | トレースプロセッサ |
| 206 | トレースデータラッチ |
| 207 | トレースデータラッチ |
| 208 | 事象カウンタ |
| 209 | チェーンラッチ |
| 210 | 性能カウンタ |
| 211 | トリガラッチ |
| 212 | トレースポートレジスタ |
| 213 | 宛先カウンタ |
| 214 | メモリ回路 |
| 215 | 伝送回路 |
| 216 | ブリスケーラ |
| 217 | 基準カウンタ |
| 218 | クロック信号 |
| 219 | 停動信号 |
| 220 | 停動済み信号 |
| 221 | トレース信号 |
| 222 | トレースデータ |
| 223 | 状態データ |
| 224 | 状態データ |
| 225 | レジスタ |
| 226 | メモリ |
| 227 | トレースバッファ |
| 300 | プロセス |
| 301 | ステップ |
| 302 | ステップ |
| 303 | ステップ |
| 304 | ステップ |
| 305 | ステップ |
| 306 | ステップ |

307 ステップ
 308 ステップ
 400 プロセス
 401 ステップ
 402 ステップ
 403 ステップ
 404 ステップ
 405 ステップ
 406 ステップ
 407 ステップ
 408 ステップ
 409 ステップ
 410 ステップ
 411 ステップ
 412 ステップ
 500 プロセス
 501 ステップ
 502 ステップ
 503 ステップ
 504 ステップ
 505 ステップ
 506 ステップ
 507 ステップ
 600 プロセス
 601 ステップ
 602 ステップ
 603 ステップ
 604 ステップ
 605 ステップ
 606 ステップ
 607 ステップ

* 608 ステップ
 701 一般的なトレースメッセージ
 702 メッセージタイプ
 703 ソースモジュール
 704 事象タイプ
 705 オーバ停動
 706 PCアブソリュート
 707 ステイタス
 708 タイムスタンプ
 10 709 ASID
 710 PCフィールド
 801 トレースメッセージの入力
 802 バッファ内の第1エントリ
 803 バッファ内の最終エントリ
 804 レジスタ
 805 レジスタ
 806 レジスタ
 807 レジスタ
 808 トレースメッセージ
 20 809 トレースメッセージ
 810 トレースメッセージ
 811 項目
 812A 項目
 812B 項目
 813 項目
 901 トレースバッファ
 902 トレースエントリ
 903 ポインタ
 1001 バッファ制御回路
 30 1002 バッファ
 *

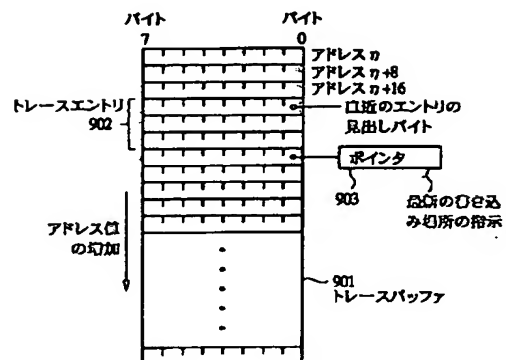
【図1】

図 1

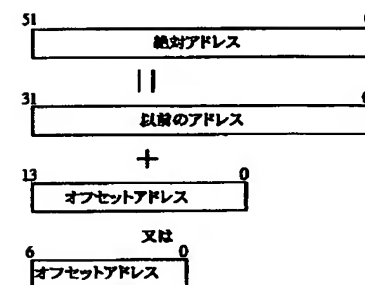


【図9】

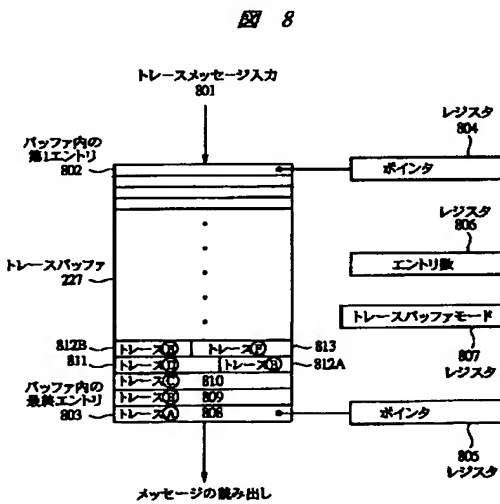
図 9



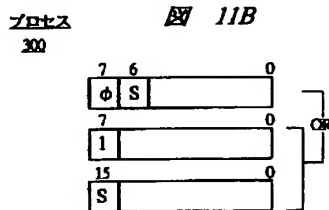
【图 11 A】



【図8】



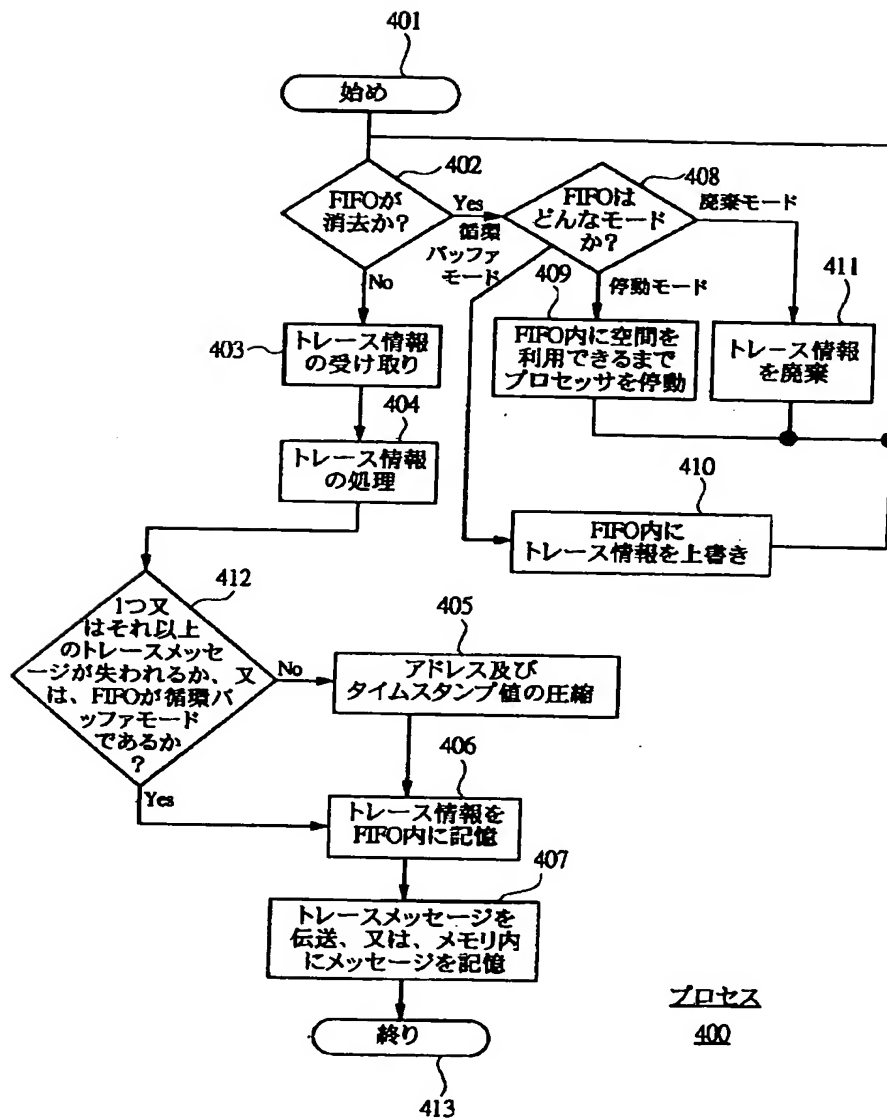
11B



オフセットアドレスのフォーマット

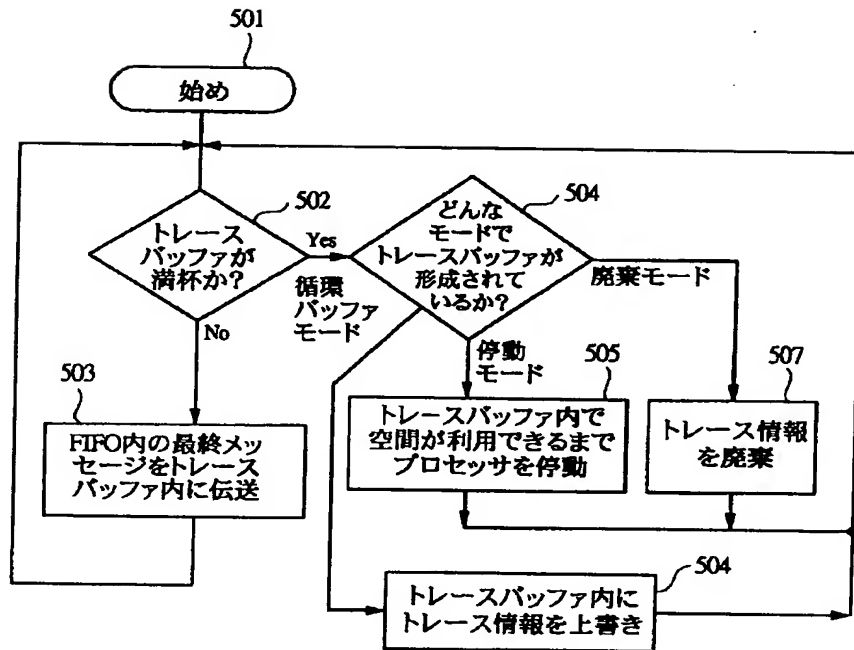
【図4】

4



【図5】

図 5

プロセス
500

【図7】

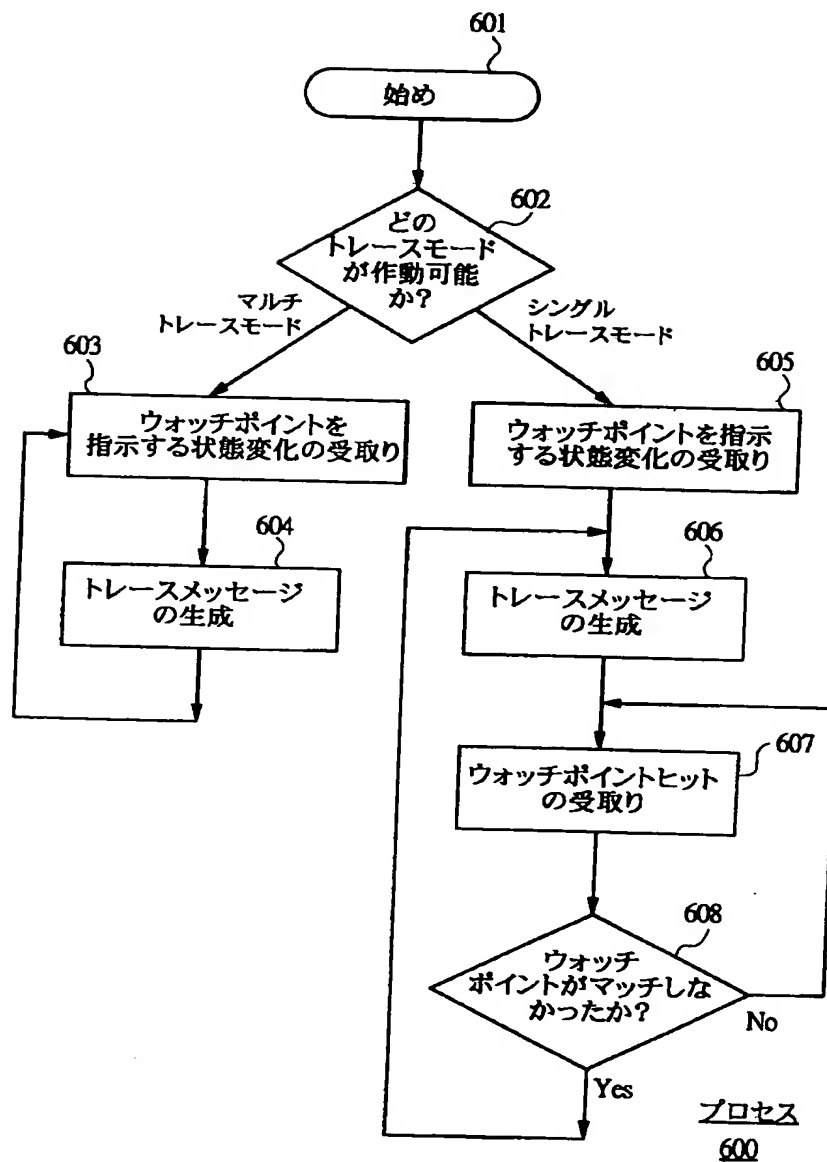
図 7

一般的なトレースメッセージ

| | | | | | | | | | |
|-----|----------|--------------|-------|----------|------------------|-----------|---------|------|---------|
| 701 | 702 | 703 | 704 | 705 | 706 | 707 | 708 | 709 | 710 |
| | メッセージタイプ | ソース モジュール | 事象タイプ | 過剰 停動 | PC アドレス リネ | ステイ タス | タイムスタンプ | ASID | PCフィールド |

【図6】

図 6



【図10】

図 10

